Forest Fire Modelling and Analytics: A Comprehensive Study

A Network Simulation and Proposal of Fire Prevention Strategies

Gian Alix
gian.alix@gmail.com
York University
Toronto, Canada

Jing Li
jliellen@my.yorku.ca
York University
Toronto, Canada

Baoqi Yu
baoqiyu@my.yorku.ca
York University
Toronto, Canada

ABSTRACT

Forest fires have been common environmental issues that have impacted millions of lives and natural resources. Thus, an in-depth study of the behavior of wildfires is critical in devising preventive strategies to control its rapid spread. However, despite a significant amount of work that has been done in the literature, developing an effective fire spread model is still a challenge.

Our goal is to design a model of forest fire propagation and be able to simulate the behavior of cascading fires in the forest. More specifically, we propose **INCINERATE**[1], a fire simulator in forest network graphs. A Linear Threshold model, together with various mathematical models from literature, were employed in this simulation that "determines" the behavior of fire propagation. We then propose fire prevention strategies in order to mitigate the damage of forest fires. In particular, we propose the **FIGHTER**[2] algorithm, a neighborhood-based edge-removal scheme utilizing the Girvan-Newman heuristic. It has been found that our edge-removal algorithm in forest networks can reduce the impact of wildfires by as much as 85% than if there were no preventive measures.

This paper discusses in-detail the models, algorithms and results of our experiments, as well as suggested further research directions.

KEYWORDS

fire propagation, forest fire analytics, information network cascades

1 INTRODUCTION

Efficient forest fire control is one of the most challenging and important problems. Wildfires have resulted in irreversible environmental and socio-economic damages across the world. According to the National Interagency Fire Center, as of November 2020, wildfires have burned more than 8 million acres in California [8]. A better understanding of fire propagation is needed to facilitate fire control and mitigate those damages. A number of models have been developed to simulate wildfire propagation in previous studies. In particular, networks have been extensively applied. In this project, we focus on simulating fire propagation through integrating topographical and weather conditions in the model.

[1] **INCINERATE** is a fire simulator we proposed and it stands for **IN**formation **C**ascades **I**n **NE**tworkX **R**epresenting **A** Forest of **T**rees **E**ngulfed by flames

[2] **FIGHTER** is a fire prevention algorithm that we have proposed and it stands for **F**orest fire **I**nhibition via a **G**irvan-Newman **H**euristic **T**hrough a neighborhood-based **E**dge-**R**emoval Approach

2 PROBLEM DEFINITION

The focus of this project is on simulating fire propagation in lattice networks (or grid networks). In particular, we aim to address the following problems:

- **Problem 1**: Given an area, construct a lattice network G, and transform elevation, slope, aspect and wind conditions into the network through linear threshold model.
- **Problem 2**: Given a lattice network, simulate wildfire propagation in the network.
- **Problem 3**: Propose fire prevention or intervention strategies to decrease the total damage caused by fire.

3 RELATED WORK

3.1 Forest Network

Hajian et al. [7] obtain the Voronoi-based network by dividing a map into many homogeneous sub-regions. In their study, sub-regions are derived by overlapping different fire environmental data layers through Geographic Information System (GIS). An inside point is assigned to represent each sub-region and Delaunay triangulation is then employed to construct the network edges. The constructed network owns significant accuracy and reality. However, most of these data layers have not been made public, and the availability of GIS applications is limited.

Based on the Cellular Automata (CA), another type of network used in fire propagation modelling is through the use of a lattice graph [9, 26]. It utilizes regularly-spaced sample points to represent landscapes. Fire propagates through the grid-cell basis. The advantage of this network is that weather conditions as well as land topography can easily be incorporated into the model [9, 26]. Pais et al. [20] partitions off the forest landscape into a series of identical area square cells. In their study, topographic data and weather conditions are integrated in the model to simulate fire growth.

3.2 Fire Propagation

Topography environment (e.g., elevation, slope, aspect) influences fire behavior quantitatively [1, 5, 12, 13, 17, 22]. In particular, the steeper the slope the faster the fire will spread [5, 12]. The aspect determines the amounts of solar radiation, moisture, and wind a slope receives. This could lead to varied temperatures and drier conditions, which can also contribute to fire behavior directly through different composition in vegetation and density [1, 5, 17]. Other research discussed burn severity being negatively correlated with elevation [13, 22].

Nelson et al. [18] analyze fire propagation and wind speed. Their experiment suggests the fire spread rate is proportional to the square of the wind speed. There are three types of fire, namely:

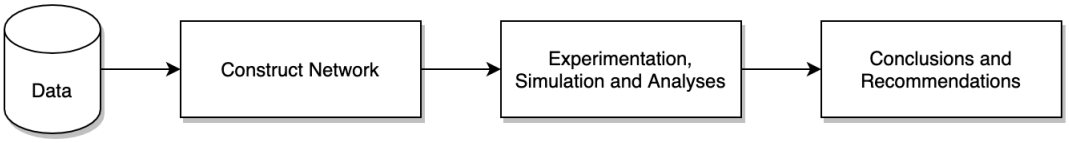


Figure 1: The high-level framework to attempt to solve the problem as discussed in Section 2 involves four major steps: (1) Acquisition and Analyses of Data, (2) Network Construction, (3) Experimentation, and (4) Conclusion and Recommendation.

surface fire, *crown fire* and *underground fire*. Under strong wind conditions, the fire will burn to the crown and the fire spread rate will increase [18]. Werth et al. [25] studies fire propagation and wind direction. They point out the spread of fire is nearly round under low wind, while under strong wind it is elliptical, and its long axis is parallel to the direction of the leading wind.

3.3 Linear Threshold Model

The Linear Threshold (LT) model has been widely used in modelling diffusion process. Pathak et al. [21] presented a generalized version of the linear threshold model for simulating multiple cascades on a network while allowing nodes to switch between them. It also has been used in social networks. Chen et al. [4] proposed a scalable algorithm to find a small set of most influential nodes under the linear threshold model. In our work, we will employ this model in the network to simulate fire propagation.

3.4 Intervention Strategies in Networks

Many studies in sociology have shown the important role of particular nodes and edges in spreading information in a complex network [27]. For example, Liu et al.[16] provided the diffusion importance of edges in his research on influence maximization. Thus, targeting these edges is significant in the planning for the control of propagation of critical pieces of information in a social or information network [10]. In epidemics, there have been several containment intervention strategies being proposed based on the idea of removing edges connecting different communities in order to efficiently contain disease spread from one community to another [3, 14].

4 METHODOLOGY

In order to address the problems as discussed in **Section 2**, we take the following steps:

(1) Secure the dataset, which is discussed briefly in **Section 4.1**.
(2) Leveraging this data, the network will then be constructed. This process will be thoroughly covered in **Section 4.2**.
(3) After the network has been constructed, we can finally perform some simulations and run some tests. **Section 5** should be able to report the detailed analyses and results of the performed experiments.
(4) Finally, we give some concluding remarks & recommendations on how the problem can be extended.

This high-level framework of our action plan can be seen visually in **Figure 1**.

4.1 Dataset

The dataset used to construct the network is the **ForestCoverType**, which contains tree observations (30×30 meter grid forest) of the Roosevelt National Forest in Colorado. Each grid includes several topographic data such as elevations, slopes and aspects [2].

4.2 Network Construction

To model wildfire propagation in heterogeneous forest landscapes, the terrain of the Roosevelt Forest in Colorado is tessellated into a number of small patches whose shape resemble squares as seen in **Figure 2** [24]. As such, the terrain is transformed into a lattice network $G(N, E)$ of dimension $|N| \times |N|$, where $N = \{v_k\}$, with $k = 1, 2, ..., |N|$ is the set of nodes (the terrain patches or the forest nodes), and E is the set of edges (links) between neighbor nodes (or neighboring forests). In the context of forest fire propagation, these links will serve as travelling routes for fire to spread from one forest to a neighboring one.

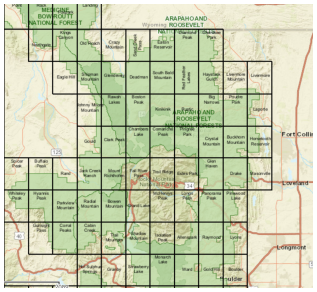


Figure 2: Grid map of the Roosevelt National Forest [24].

Each node in the network graph has a state serving as one of its node attributes. At time t, a node can be in any one of the following states: *empty*, *not_burned*, *burning*, or *burnt*. The "empty" represents areas where there are no trees (cannot be burned). The state "not_burned" represents trees that are intact. The states "burning" and "burnt" represent trees currently burning and have been burnt down respectively. A user-defined parameter ρ then determines the forest density of our network, i.e. this density-factor controls the percentage of the total areas in the network containing forests.

The deciding factor in the fire propagation process of the simulation are numbers as computed by a linear threshold model (discussed in **Section 4.3**). Each node v_k uses three topographic features, namely the elevation, slope and aspect, as factors to compute the node's threshold θ. This indicates how easily v_k would switch from one state to another. Each edge (v_k, v_l) is weighted and the weights β can be calculated using wind and Euclidean distance factors. Thus, the spatial distribution of fire propagates is converted to a cascading problem in networks.

4.3 The Modified Linear Threshold Model

A modified version of the LT model as discussed in **Section 3.3** was employed in the simulation of the forest fire propagation. In this diffusion-based network model, nodes can have two possible states: *active* and *inactive*. Therefore, in the context of our forest network, active nodes are those forests that are currently burning (i.e. having 'burning' state) while inactive ones are those that have not yet been burnt (i.e. having 'not_burning' state). Clearly, those nodes in the system that are quoted as 'empty' have no role to play in this LT model. And those burnt out forests can be thought of as 'inactive' in the LT model as fire has already been extinguished and can no longer influence neighboring forests. However, these burnt nodes, while considered inactive, 'loses' their role to play in the LT model as burnt out forests cannot go back to the burning state.

Each node then in the network has some uniformly random $\theta \in [0, 1]$ which serves as a threshold for when its state flips. One tweak proposed by our team is that we can change the way we pick these uniformly selected random numbers. A mathematical model to compute these node thresholds is described in detail in **Section 4.4**. Then each edge in the (undirected) network will have some randomly assigned weight $\beta \in [0, 1]$. Similar to node thresholds, we employ a more mathematical-based formula to calculate edge weights as detailed in **Section 4.5**. Thus, for any inactive node (or 'not_burning' forest) v in the forest network, its state will switch to an active (or burning) state if and only if the total weights of all edges of e_{uv} is greater than the node threshold of forest v, where u is an active (burning) neighbor of v. [11] Mathematically speaking:

$$\left(\sum_{u \in [\text{active neighbor of } v]} w_{u,v} > \theta_v \right) \implies \text{switch state of } v \quad (1)$$

The process initially begins with all nodes being inactive. A random node in the network is then selected to be active and the diffusion process happens for that node (cascading to its neighbors). To translate this in the context of our forest network system, we begin with an initial setup where no forests are burning. Then a forest is selected at random, which serves as the ignition source of the forest fires. Forest fire propagation should then begin to take into effect and will cascade to its neighbors and to its neighbors' neighbors, according to the mathematical process as described by **Eq. (1)**. This cascading process is repeated and iteration terminates when any of the following becomes true:

- If we have reached the maximum number of timestep iterations, which can be parameterized
- When all inactive nodes in the network have been switched to active, or in this case if all non-burning forests have turned burning or have become burnt
- When there are no new nodes or forests that have become active or have been switched to the burning state from the previous iteration

The algorithms as discussed in **Section 4.7** should give a better understanding of how the linear threshold model can be implemented in the simulation of forest fire propagation.

The tweak that the team proposed to modify the method for selecting node thresholds and edge weights is rooted from the limitation of the basic LT model. In the basic model, node thresholds and edge weights are selected uniformly at random. Here, the rationale for this choice of method of picking random values was the lack of knowledge of the network, as identified by Kempe et al. [11]. However, as we have additional knowledge on the context of the network that we are modelling, then it may be imperative to consider the various fire spread factors. This involves designing mathematical models that can compute the node thresholds and edge weight values. The LT model itself is designed to handle the fire cascading effects, and so the only concern then is to model the values for node thresholds and edge weights.

4.4 The Mathematical Formulation of the Node Thresholds

In this section, we attempt to discuss thoroughly the mathematical model behind the formulation of the nodes' thresholds. Each node in the network graph is assigned node threshold θ, which can be calculated as:

$$\theta = -\frac{1}{\pi} \arctan(\phi_s \xi \alpha) + 0.5 \quad (2)$$

Table 1 presents us with a tabulated summary of the node threshold symbol, nomenclature, and formula for such parameters. For instance, the slope coefficient ϕ_s can be obtained using the formula

$$\phi_s = 5.275(\tan \phi)^2 \quad (3)$$

where ϕ is the slope found in the **ForestCoverType** dataset [2]. The direct relationship between slope steepness and likelihood of fire is dictated in this formula and that this is actually from a portion of Rothermel's fire spread model [23]. In fact, many pieces of various existing fire propagation math models were used and incorporated in our mathematical models. Another example can be seen in the formula for calculating the elevation coefficient ξ, which was borrowed from Olabaria, et. al. [19] and can be calculated by:

$$\xi = \frac{1}{1 + \ln(\max\{he^{-6}, 1\})} \quad (4)$$

where h is the elevation value and again can be found in the **ForestCoverType** dataset [2]. It can be noticed that the elevation & likelihood for fire are negatively-correlated. This means that the higher the elevation values of a forest, the lower its risk for it to catch on fire. The rationale behind such claim is that higher elevation lands tend to receive less forest fuels than those in lower-elevated places. As a result, higher places are less prone to catch fires than those in the lower areas. Hence, this should clearly explain why elevation presents itself to be an inverse factor in the mathematical model, which was also observed by Olabaria, et. al [19].

We then turn our attention to the aspect coefficient α. The study by Estes et al. [5] provides us with information on how we can obtain the aspect coefficient α based on the aspect value A. It was also be observed by Estes, et. al. that eastern aspect directions (with $A \in [22.5, 157.5)$) has higher aspect coefficients α, as indicated in the lookup table **Table 3** from **Appendix C.4**. Again the aspect values can be found in the **ForestCoverType** dataset [2], whose values range from $A \in [0, 360]$.

In the math formula for the node threshold found in **Eq. (2)**, we can see that the function applied is $\arctan(\cdot)$ and the choice of such function is arbitrary. In fact, any function would have worked so

Table 1: Model Parameters used for the Node Threshold θ and the Edge Weight β

Symbol	Nomenclature	Formula	Notes
ϕ_s	slope coefficient	$5.275(\tan\phi)^2$	Rothermel [23]; ϕ is the slope in the ForestCoverType dataset [2]
ξ	elevation coefficient	$\frac{1}{1+\ln(\max\{he^{-6},1\})}$	Olabarria, et. al. [19]; h is the elevation in the ForestCoverType dataset [2]
α	aspect coefficient	(See **Table 3**)	Estes, et. al. [5]; See **Appendix C.4** for the lookup table
ϕ_w	wind speed	$\gamma\cos\tau$	Calculated; $\gamma = \text{RANDOM}[0,\psi]$ where ψ is user-defined; τ given in **Eq. (7)**
δ	node Euclidean distance	$\sqrt{\Delta^2 x + \Delta^2 y}$	Calculated: Δ denotes the horizontal (x) and vertical (y) node distances

as long as the function is monotonous. The range of the values of $\arctan(\cdot) \in [-\frac{\pi}{2}, \frac{\pi}{2}]$, which should clearly explain the purpose of $\frac{1}{\pi}$ that is merely used as a scaling factor. Hence this particular constant can change depending on the choice of function used in place of $\arctan(\cdot)$. Furthermore, the additional 0.5 constant should give an idea that it is a correction factor as the first term has range of values from $[-0.5, 0.5]$. Hence, our mathematical model for node threshold should be some number between $[0, 1]$.

4.5 The Mathematical Formulation of Edge Weights

Like the model used for formulating the node thresholds, we also have a mathematical model to compute the edge weights in our network. The weight β of the edges can be computed as follows:

$$\beta = \max\left\{\frac{2}{\pi}\arctan\left(\frac{\phi_w}{\delta}\right), 0.01\right\} \tag{5}$$

where the symbol, nomenclature and formulas can be found in **Table 1**, similar to where the node threshold parameters are tabulated. The parameter ϕ_w is the wind speed and can be expressed as a function of the wind magnitude γ and wind direction τ:

$$\phi_w = \gamma\cos\tau \tag{6}$$

The wind direction τ (in degrees) can easily be computed using relative locations on the forest map, which we define as:

$$\tau = \begin{cases} \arctan(\Delta y/\Delta x) & \text{if } \Delta x \neq 0 \\ \text{sgn}(\Delta y)\cdot 89.9^{\text{o}} & \text{otherwise} \end{cases} \tag{7}$$

This piecewise function avoids the division of zero problem when $\Delta x = 0$ and further ensures that $\cos\tau \neq 0$. The wind magnitude on the other hand is a random number between $[0, \psi]$ where ψ is a user-defined parameter that determines the max possible wind speed at any given time t. Finally the Euclidean distance δ, also based from the positions of the nodes in the network, is a another factor considered that should reduce the influence of fire propagation for whenever there is a large distance between any two forests.

We have made a unanimous decision that edge weights cannot be zero as that brings forth no effect towards fire propagation, for any burning node. This is unrealistic and it may cause problems during our simulations. As such, we ensured that if $\phi_w = 0$ causing $\arctan(\cdot)$ to be zero, then we simply set $\beta = 0.01$. Furthermore, we use the $\max\{\cdot,\cdot\}$ function to satisfy $0.01 \leq \beta \leq 1$.

4.6 Fire Prevention Strategies

After a simulation of the forest fire propagation (to be discussed in **Section 5**), we desire to devise some schemes for how we could control the fire spread, or at least decrease the total fire damage. There are several ways to do so, such as through fire intervention, fire prevention, etc. In our research project, we aim to focus on strategies to prevent fire[3]. We could do either an edge-removal or node-removal in our forest network. In the context of our forest graph, removing nodes entail chopping down of trees or forests. To avoid the cause of harm to the environment, we opt to turn our focus on the removal of edges or cutting off links between forest nodes, i.e. cutting off the path for where fire can travel. In our study, we discuss some edge-removal strategies to target fire reduction in our forest network (than if no prevention method was applied).

In our (incomplete) forest network, some (forest) nodes are easily reachable from another; while not-so for others. The configuration of the graph and the connectivity of the nodes play a big role in reachability from one forest node to another. Because of this, we can look at schemes that relate to shortest-path properties of the forest graph. In particular, we may be interested in the graph's edge-betweenness. Recall the *edge-betweenness* of edge e_{ij} to be the number of node pairs (u, v) such that edge e_{ij} lies on the shortest path between nodes u and v. As there can be several shortest paths between u and v, then the betweenness of e_{ij} is the fraction of those shortest paths that include e_{ij} [15]. Betweenness can be further used to detect communities, as discussed in the literature by Girvan, et. al., and an algorithm they proposed is further described in detail in **Appendix A.2** [6]. Using the Girvan-Newman (GN) algorithm in the edge-removal scheme for fire prevention is especially beneficial in that (1) we are removing edges with the highest betweenness scores (i.e. the paths for where fire are more likely to travel across), and (2) isolating communities of forests (i.e. any removed "bridge" prevents fire spread from one community to another).

The problem that can be foreseen in the use of GN algorithm to counteract fire propagation is that although this scheme removes edges with the highest betweenness scores that translate to paths of higher likelihood for fire spread, it might not necessarily be the case that such edges propagate through those paths. For instance, an inactive node being considered to switch states because at least one of its neighbors is burning does not mean that it will necessarily flip its state. This is especially if its threshold is significantly high

[3]Fire prevention deals with decreasing fire damage before fire could happen; fire intervention does the same except in real time, when fire is already occurring.

Algorithm 1: INCINERATE ($G(N, E)$)

Input : The forest graph G, with forest set N and probable fire path set E

Output : (None). The resultant forest graph of the cascading Linear Threshold model

```
1  Γ ← {f | f ∈ N ∧ f.fire_state = 'burning' }
2  foreach v in Γ do
3      Λ ← v.neighbors
4      if Λ = ∅ then
5          break
6      foreach n in Λ do
7          Ψ ← {t | t ∈ n.neighbors ∧ t.fire_state = 'burning'}
8          s ← 0
9          foreach u in Ψ do
10             s = min(1, s + β_{u,v})
11         if s > θ_n then
12             n.fire_state ← 'burning'
```

enough and that its adjacent edges have significantly low valued-weights. Another possibility is that the sequence of edges chosen by GN might not be appropriate enough to cause a significant amount of impact in impeding fire spread; in that a location-based scheme would have been a critical criterion to consider. For instance, if GN chooses an edge to remove that is located somewhere close to the lower-leftmost part of the map as it has the highest betweenness and that recalculating the betweenness scores causes GN to find the new edge with the largest betweenness in the upper-rightmost area of the map, then this definitely does not result into an improvement of hindering fire propagation as the two successive edges removed are distant of each other. Thus, we desire a method that chooses edges that take location into account. In particular, we wish to select an edge with a high betweenness score that is also proximate to the edge that was previously pruned. For this, we propose the **FIGHTER** algorithm and how it works is discussed further in **Section 4.7.2**. Throughout the discussion of the proposed intervention algorithm, we define a few terminologies, such as the *edge distance*.

Definition 4.1 (Edge Distance). The edge distance Δ between two edges e_{ij} and e_{uv} in the forest network is defined to be:

$$\Delta(e_{ij}, e_{uv}) = \min\{\delta_{i,u}, \delta_{i,v}, \delta_{j,u}, \delta_{j,v}\} \tag{8}$$

where $\delta_{x,y}$ is the Euclidean distance between nodes x and y. In other words, Δ is the shortest distance between one node of an edge and another node of the other edge.

The notion of the edge distance will allow us to reason about the location-based edge-removal in a forest network. Furthermore, it is also critical to define what it means for an edge to be in the (local) neighborhood of another edge. An edge e' residing in the neighborhood-ball of an edge e that recently got pruned out of the network is a necessary criterion in our edge-removal algorithm.

Definition 4.2 (Neighborhood-Ball of an Edge).
The neighborhood-ball of an edge e, denoted as B_λ, for a parameterized $\lambda \in \mathbb{R}^+$, is the region in the forest network consisting of all edges e' that are within an (edge) distance of λ from e, i.e.

$$B_\lambda(e) = \{e' \mid e' \in E \wedge \Delta(e, e') < \lambda\} \tag{9}$$

Algorithm 2: SIMULATE-FIRE ($G(N, E), t$)

Input : Graph G with set of nodes N and set of edges E, and the maximum iteration number t

Output : (None). The resultant G with fire simulated

```
1  u ← pick a random forest ∈ N
2  u.fire_state ← 'burning'
3  B ← {u}; B̂ ← ∅; k ← 1
4  while k ≤ t ∧ B ≠ B̂ do
5      INCINERATE(G(N, E))
6      B̂ ← B; B ← {f | f ∈ N ∧ f.fire_state = 'burning' }
7      k ← k + 1
```

4.7 The Algorithms

In this section, we briefly discuss the main algorithms, **INCINERATE** and **FIGHTER**, that were employed to solve the problem.

4.7.1 INCINERATE: Fire Simulation Algorithm

We first turn our attention to **Algorithm 1** and **Algorithm 2**. It can be observed that the **INCINERATE** algorithm is simply a subroutine of the main SIMULATE-FIRE method. It can also be observed that SIMULATE-FIRE is simply the procedure that was discussed in **Section 4.3**. In Lines 1-2, some random forest in N is selected at random and this will serve as the ignition source of fire in the network. In Line 3, we introduce three new variables that represent the set of currently burning forests B, the set of burning forests in the previous timestep $\hat{B}$, and the iterator k. The cascading process then comes into play in Lines 4-7 where the stopping criterion is when either the max number of iterations has been reached (parameterized by t; $t = \infty$ for no specific max iterations), or when the set of currently burning forests is the same set of burning forests in the previous timestep (i.e. no new forests have switched to the burning state). Exploring the `while` loop, in Line 5 we perform the **INCINERATE** procedure of the graph G (see below for details). Then in Line 6, we let the current set of burning forests to be the set of forests burning in the previous timestep. And then obtain the new set of currently burning forests. Then increment our iterator in Line 7 to prepare for the next iteration of the loop.

We then have a look at the **INCINERATE** sub-algorithm. In Line 1, Γ will contain all those forest nodes that are burning. We then operate a `for-each` loop in Lines 2-12, where for each burning node v in Γ, we do Lines 3-12. We gather all the neighbors of forest v and store them into Λ. In Lines 4-5, we check if the burning forest v does not have neighbors; in that case, we simply break off the loop. Otherwise, in Lines 6-12, we take each of those neighbor nodes of v that was stored in Λ. And then, for each node n in Λ, we obtain all neighbors of n that are burning and store them in Ψ as seen in Line 7. Lines 8-10 computes for the total sum of the weights of the adjacent edges of n and caps it off at 1, to keep the values within $[0, 1]$. In Line 11-12, we check if the sum of those edges surpass the node threshold of node n. If this is the case, then the forest node n switches to the 'burning' state. This is what is happening under the hood of **INCINERATE**, which should give a clearer picture of the main method SIMULATE-FIRE.

A sample simulation that employs these algorithms can be seen in **Figure 7** of **Appendix C.1**. Greater discussion can be found in

Algorithm 3: FIGHTER $(G(N,E), K, \lambda)$

Input : Graph G with set of nodes N and set of edges E where $|E| \geq 1$, K number of edges to remove (an integer such that $0 < K \leq |E|$), and threshold $\lambda > 0$

Output : E', the edges removed based on edge-betweenness centrality and local neighborhood. The resultant G is now G' where $G' = G(N, E \setminus E')$.

```
1  CALCULATE-EDGE-BETWEENNESS(G)
2  e_max ← MTH-LARGEST-EDGE(G, 1)
3  B_λ ← {e | e ∈ E ∧ EDGE-DISTANCE(e_max, e) < λ}
4  E ← E \ {e_max}; E' ← {e_max}
5  CALCULATE-EDGE-BETWEENNESS(G)
6  m ← 1
7  while |E'| < K ∧ m ≤ |B_λ| do
8      e_m ← MTH-LARGEST-EDGE(G, m)
9      if EDGE-DISTANCE(e_max, e_m) < λ then
10         B_λ ← {e | e ∈ E ∧ EDGE-DISTANCE(e_m, e) < λ}
11         E ← E \ {e_m}; E' ← E' ∪ {e_m}
12         e_max ← e_m; m ← 1
13         CALCULATE-EDGE-BETWEENNESS(G)
14     else
15         m ← m + 1
16 return E'
```

Section 5. Furthermore, an equally important, yet less relevant algorithm is the SIMULATE-WIND algorithm, which factors in the wind component. See **Appendix A.1** for a more in-depth exploration of the algorithm and **Appendix C.3** for a sample simulation.

4.7.2 FIGHTER: Fire Prevention Algorithm

In this section, we cover the **FIGHTER** algorithm, a neighborhood-based edge-removal algorithm based on betweenness centrality scores. We have discussed earlier in **Section 4.6** that this algorithm takes its inspiraton from a Girvan-Newman heuristic where we sequentially take off edges with the highest edge-betweenness scores (while recalculating betweenness scores for whenever an edge is pruned from the network). However, we did see some problems with the GN approach. Hence, we propose an algorithm that takes into account the local neighborhood of edges being removed from the network as an additional basis for selecting edges to remove.

Referring to **Algorithm 3**, we do a somewhat similar approach to GN. We first calculate the edge betweenness scores of all edges in G in Line 1. Then in lines 2-5, we take the edge in G that has the highest betweenness score and then assign this to $e_{\max}$[4]. We then obtain the neighborhood-ball of this edge $e_{\max}$, as how we defined a neighborhood-ball in **Definition 4.2**. We then remove this edge from the set of edges and then create a set or list of edges that have already been removed in E (called E'). In line 6, we let $m = 1$; we will increment this number in the `while` loop body of lines 7-15 whenever we cannot take out the edge that has the mth largest edge-betweenness (due to the location-based factor that was not satisfied). Otherwise, then we can reset $m = 1$ again as edge betweenness for all edges in the network graph are recalculated.

[4]Note that MTH-LARGEST-EDGE(G, m) takes the edge in G with mth largest edge betweenness score.

Examining the loop body, there are two terminating conditions: (1) when $|E'| \geq K$ (i.e. the number of edges noted down already exceeds the K number of edges the user asked to remove), and (2) is when $m > |B_\lambda|$, which indicates that there are no more edges in the neighborhood-ball of the most recent edge removed. So by (2), it is clear that it is possible that the **FIGHTER** algorithm can remove less than K edges as requested by the user. In line 8, we obtain the edge that has the mth largest score based on betweenness centrality. Then we check if this edge resides in the neighborhood-ball of the previously pruned edge. If not (lines 14-15), then we check the edge with the next largest edge-betweenness score (and keeping doing so until we find the next edge having the highest betweenness that is resiging in the neighborhood-ball). If it is (lines 9-13), then we obtain the new neighborhood-ball, remove the edge from E and then add it onto the constructed set E' of removed edges. In line 12, we update what was the most recent edge pruned from the network and then reset $m = 1$. Finally we have to recalculate the edge-betweenness for all edges as we pruned an edge from the forest network. In line 16, simply return the set of edges removed. By then, the graph G should have K (or less) edges removed, based on the betweenness centrality scores and also on the basis of the pruned edges' local neighborhood.

4.7.3 Benefits and Limitations of the Algorithms

INCINERATE is a simple, straightforward simulator for propagating fire. However, it employs a decision-based cascading networking model that is deterministic (i.e. its thresholds and weights are calculated based on math formulas as opposed to stochastic), and cannot be compared to the stochasticity of fire in real life.

FIGHTER (and GN) are also straightforward schemes that aids in the prevention of fire. They are effective as they aim to remove graph edges where fire is most likely to travel and we will see in **Section 5** that they work well. However, such hypothetical methods make it impossible to compare the models in the real world forest network. Fire is irreversible and stochastic.

5 EXPERIMENTS / EVALUATION

5.1 The Network

The number of nodes in our network is parameterized. However, we will be using the standardized $|N| = 100$ nodes for our experiments unless otherwise specified[5]. Each node also has a chance to have an edge with its adjacent nodes with probability ρ, the network's density factor, which is standardized at $\rho = 0.8$.

5.2 Experiments and Analyses

We have tested and run our simulations around 100 times. A sample sequence of the simulations in **Figure 7** of **Appendix C.1**.

5.2.1 Forest Damage Overtime

We have performed some experiments and simulations. Firstly, we looked into the percentage of forests in the map that are both burning and burnt down overtime. A plot of this can be seen in **Figure 3** and our team observed some interesting results. For one, which may seem intuitive is that the percentage of damaged forests (which

[5]Some simulations require more nodes to see more interesting analyses. In such cases, we'll use $|N| = 400$ nodes.

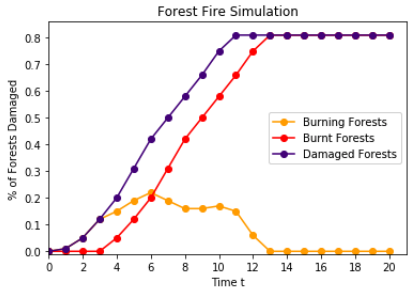


Figure 3: This plot shows the percentage of the area damaged overtime: the fraction of those forests burning, burnt down, and those damaged (note that the total damage is simply the number of forests that are burning and the number of forests that have been burnt.). $|N| = 400$ nodes in this case.

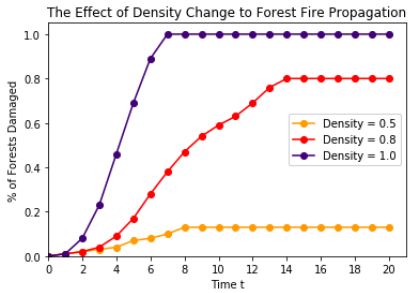


Figure 4: This plot shows the percentage of forests damaged overitme for different values of the forest density factor ρ. This includes both those forests that are burning and those that are burnt. $|N| = 400$ nodes in this case.

is simply the total number of burning and burnt forests) tends to converge to ρ, the density factor of the system. This suggests that all (or perhaps almost all) of the forests will eventually be impacted by fire, especially for higher values of ρ that implies a more connected forest network. Details on the analysis of ρ will be discussed in **Section 5.2.2**. Another thing to notice is that the number of burning forests would initially be greater than the number of burnt forests in the system until at some point before the number of burnt forests will be larger. This we saw was always true and that there seems to exist a time step threshold t' such that past this point, then the number of burnt forests is greater than the number of of burning forests. Another clear observation is the shapes of the plots for the number of burnt and damaged forests. They seem to follow the same shape patten, with the exception that the number of burnt forests is simply shifted to the right. And lastly, the percentage of the burning forests will always have this shape: that is, it will increase until it hits some threshold (say ω) and then past this ω, it will either decline or it will plateau then decline. Afterwhich, it will stagnate to 0 until the end of time. These are clear observations and pretty much intuitive. Fire spreads, propagates and then dies.

5.2.2 The Density Factor of the Forest Network (ρ)

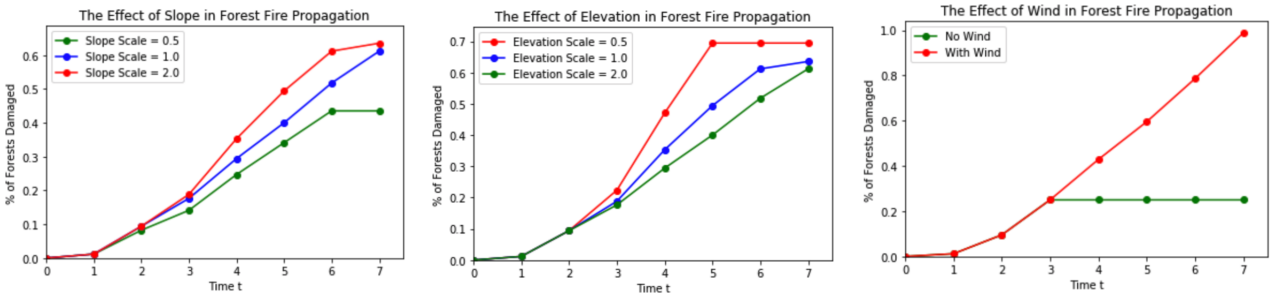
The observations and analyses conducted from **Figure 3** are more or less the same for all simulations we have tested on. However, we did notice changes for when we cranked up the density factor ρ. In **Figure 4**, we have attempted using various ρ values. Aside from $\rho = 0.8$, we also used $\rho = 1.0$ and $\rho = 0.5$. For the situation for when $\rho = 1.0$, this suggests that the entire system is densely forested and that all neighboring nodes/forests are connected together. As a result, fire propagates at a much faster rate and it can be observed from the plot that as early as time $t = 6$, the entire forest system has already burnt down. And we can see that this is not the case for the other ρ values. When $\rho = 0.8$, it takes 14 time steps before the entire forest system burns down. However, it is a different case for when $\rho = 0.5$. In this setting, roughly only half of the system is forested. This implies that some forests may be disconnected from others due to a great number of fire zones or areas that are not burnable. As a result, forest fires do not propagate easily to all areas and from the plot that we see in **Figure 4** that roughly only 10% of those forested areas have been damaged. It does not also support the claim that the percentage of damaged areas approach ρ in the long run as this is not the case for when $\rho = 0.5$. Perhaps this is the case for all scenarios for when $\rho > \rho'$ but not so for when $\rho \leq \rho'$ (such as when $\rho = 0.5$), for some ρ'. Therefore, the plot as displayed in **Figure 4** verifies our hypothesis that the more connected and the more dense the forest network is, the more likely it is for fire to spread. Additionally, fire propagates at a much faster rate and more forests will thus get destroyed at an earlier time point. Furthermore, this implies that the density factor ρ indeed does play a big role in the simulation of spread of forest fires.

5.2.3 The Effect of Slopes on Forest Fire Propagation

From the literature, it has been observed that a higher slope implies that it is more likely that fire propagates and at a much faster rate. Let us test this. If we simulate the fire propagation three times (the same starting graph initialization and configuration including ignition source origin), keep all other factors constant and simply changed the slope values for each of those runs (one will be the original slope, the next simulation will have slopes that are twice as large as the original, and the last simulation having slopes with half as large as the original), then we can see from the result in **Figure 5(a)** that the observation from the literature has been validated. This is because, overtime, we can see that the large-scaled slope (red plot) produces more fire damage to the forest network. However, for forests on lands with lower slopes (the green plot), we can observe that the damage by fire propagation is not as great as the damage for when the slope is larger. Another interesting observation is that up to some certain timestep t', the amount of fire damage to the forest are almost approximately the same. In this case $t' \sim 2$.

5.2.4 The Effect of Elevations on Forest Fire Propagation

Based from the literature, it was observed that there was indirect relationship between elevation and fire likelihood, i.e. higher elevation values indicate lower risk for fire propagation. We shall test this. In **Figure 5(b)**, we can see the results of our experiment. Here, similar to the slope testing, we did the same for elevation (with $\times 0.5$, $\times 1.0$ and $\times 2.0$ scales to the original elevation). The experimental results validate the observation from literature where lower elevation causes fire easily (more damage caused by fire as shown by the red plot for half the original elevation value). The green plot (double the original value) also indicates less damage to



(a) (b) (c)

Figure 5: The above shows a plot overtime of the percentage of forests damaged for different (a) slope values, (b) elevation values, and (c) presence of wind. We note that in testing the effect of one variable, we will keep other factors constant.

fire, as found in the literature. It is also interesting to note, like the slope factor, that the damage caused by fire is almost approximately the same from the initial time up until some time point t'. In our plot in **Figure 5(b)**, this would be about $t' \sim 3$.

5.2.5 The Effect of Wind on Forest Fire Propagation

By intuition (and observation of how fire behaves), wind brings about great impact on the behavior of fire. Putting this onto the test in our simulator, let us compare the simulation for when there is wind component versus the simulation when wind is taken out of the picture. The results are plotted in **Figure 5(c)**. Notice how wind can bring forth drastic damage to the forest by the fire propagation (i.e. almost the entire forest network is burned). This is seen in the red plot. On the other hand, the green plot which represents no wind component, only has around 20-25% of the forest network is damaged by fire. We can also observe that without wind, the fire propagation process terminates and stablilizes at an early time $t = 3$. Also, it is interesting to see that whether there is wind or no wind from time $t = 0$ up until some time $t' = 3$, then the amount of damage is not-so different. This is purely coincidental as wind is varying (randomly) every T timesteps (the simulation here could possibly have $T = 2$ perhaps).

5.2.6 Performance Analysis of Fire Prevention Strategies

The performance of our proposed **FIGHTER** algorithm is compared with that of the performance of the benchmark algorithm of Girvan-Newman. We also compare these algorithms with the case for when no prevention methods were applied. In our simulations[6], we take a sample of 5 runs which we display as seen in **Table 2** (we use the same set of parameters in all 5 runs). It can be observed that in all instances, applying the **FIGHTER** algorithm can decrease the total fire damage to the forest network by the end of the fire propagation simulation, than if no prevention algorithm was applied. Girvan-Newman's algorithm also (slightly) improved the condition for where there was no prevention at all, i.e. it either did better or the same but it never performed worse. A significant observation that can be seen from **Table 2** is that in 4 out of the 5 times of simulating fire propagation, our proposed **FIGHTER** algorithm does at least as better than the Girvan-Newman approach. This is an indication that the method that we propose gives some promising results. In particular, Simulation 1 in **Table 2** shows the best results performed by our **FIGHTER** algorithm, in that it was able to reduce the total fire damage by about 85% than if there was no fire prevention strategy applied. Our edge-removal algorithm for this particular simulation forced an early stopping for the fire propagation process, stabilizing the conditions as early as time $t = 3$, compared to when no fire prevention is at play at time $t = 8$. Comparing this improvement brought by the Girvan-Newman heuristic where there was only about 15% reduction rate to the total fire damage in the forest network. We can see a plot,

Parameters: $K = 5, \lambda = 75$ meters

Prevention Methodology	Number of Timesteps before Termination	Total Fire Damage (%)
None	8	65.9%
Girvan-Newman	8	55.3%
FIGHTER	3	9.41%
None	7	61.5%
Girvan-Newman	7	48.8%
FIGHTER	7	48.8%
None	5	57.6%
Girvan-Newman	5	57.6%
FIGHTER	5	40.0%
None	6	67.1%
Girvan-Newman	6	22.4%
FIGHTER	6	67.1%
None	6	96.7%
Girvan-Newman	6	96.7%
FIGHTER	6	83.0%

Table 2: Results of a sample of five simulations for the various methods of fire prevention strategies. The algorithm highlighted in yellow gives the best performing prevention scheme for each simulation.

[6] A prevention strategy A is better than method B if the total fire damage after a simulation for when A is applied is less than the total fire damage of B for when applying B to the network. The same forest network configuration and ignition source is used during simulations. Apart from fire damage criterion, we also check for the number of timesteps before the process terminates, as seen in **Table 2**.

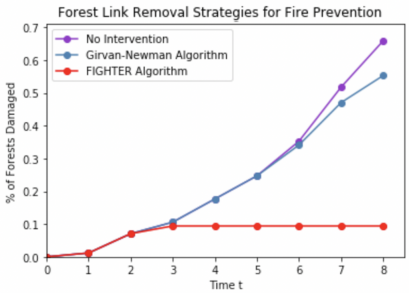


Figure 6: This plot shows the percentage of forests damaged overtime based on the various fire prevention strategies discussed. Note that configuration of the initial forest network (including the ignition source) are all the same across all schemes. This is the plot for Simulation 1 in Table 2.

displaying the performances of these algorithms on this particular simulation in **Figure 6**. Based on the results of our simulations, we came up with some theorems. The first one in particular relates to the set of edges as removed by **FIGHTER** and by the GN.

Theorem 5.1. Let G be a forest network graph. Let G_F be the forest network graph after removing a set of edges $\mathcal{F}$ from G according to the **FIGHTER** algorithm. Let G_G be the forest network graph after removing a set of edges $\mathcal{G}$ from G according to the Girvan-Newman approach. Then $\mathcal{F} = \mathcal{G}$ if and only if the following hold:

(1) $G_F = G_G$
(2) The performance of both intervention strategies are the same

A short proof of these algorithms is written on **Appendix B**. Now, we introduce another theorem as a result of our findings in our experiments. This theorem relates to the neighborhood-ball threshold radius λ parameter in our proposed method.

Theorem 5.2. If $\lambda \to \infty$, i.e. the neighborhood-ball of any edge covers the entire network space, then $\mathcal{F} = \mathcal{G}$.

6 CONCLUSIONS

In the simulation of forest fire propagation, we begin by constructing a lattice network while incorporating topographic features such as elevation, slope and aspect in our networks. Our fire simulator, **INCINERATE**, works under the modified Linear Threshold model. We then look into some strategies for fire prevention, such as the use of Girvan-Newman's approach and our proposed **FIGHTER** algorithm, both of which rely on betweenness. Additionally, our proposed method relies on (local) neighborhood-based criterion.

A possible future work could involve examining how the hyperparameters K and λ are tuned in our proposed method. We do not wish to choose a K that is small enough that it brings little to no impact on the fire propagation; nor select a K large enough that disconnects the majority of our forest network. We also hope to be not too strict by choosing small λ, in that there may only be a limited number of edges to choose from within the neighborhood-ball that have high betweenness scores. However, we also hope to be not too lenient that we consider large λ values where the neighborhood-ball considers the entire forest network (thus reducing to the GN algorithm by **Theorem 5.2**). Additionally, we may want to investigate intervention methods as well.

GITHUB REPOSITORY

Visit `https://github.com/techGIAN/ForestFireAnalytics` for the full code of our project.

MACHINE SPECIFICATIONS

All experiments were conducted on a MacBook Pro with Dual-Core Intel Core i5 CPU @ 2.3 GHz and 8GB memory, using Python 3.7.4. All models are generated by the `NetworkX` package. Experiments and results are tested and validated using a MacBook Air with Dual-Core Intel Core i5 CPU @ 1.6 GHz and 8GB memory.

REFERENCES

[1] Patricia M Alexandre, Susan I Stewart, Miranda H Mockrin, Nicholas S Keuler, Alexandra D Syphard, Avi Bar-Massada, Murray K Clayton, and Volker C Radeloff. 2016. The relative impacts of vegetation, topography and spatial arrangement on building loss to wildfires in case studies of California and Colorado. *Landscape Ecology* 31, 2 (2016), 415–430.

[2] Jock Blackard, Denis Dean, and Charles Anderson. 1998. Covertype Data Set. https://archive.ics.uci.edu/ml/datasets/covertype

[3] Per Block, Marion Hoffman, Isabel J Raabe, Jennifer Beam Dowd, Charles Rahal, Ridhi Kashyap, and Melinda C Mills. 2020. Social network-based distancing strategies to flatten the COVID-19 curve in a post-lockdown world. *Nature Human Behaviour* (2020), 1–9.

[4] W. Chen, Y. Yuan, and L. Zhang. 2010. Scalable Influence Maximization in Social Networks under the Linear Threshold Model. In *2010 IEEE International Conference on Data Mining*. 88–97. https://doi.org/10.1109/ICDM.2010.118

[5] Becky Estes, Eric Knapp, Carl Skinner, Jay Miller, and Haiganoush Preisler. 2017. Factors influencing fire severity under moderate burning conditions in the Klamath Mountains, northern California, USA. *Ecosphere* 8 (05 2017). https://doi.org/10.1002/ecs2.1794

[6] M. Girvan and M. E. J. Newman. 2002. Community structure in social and biological networks. *Proceedings of the National Academy of Sciences* 99, 12 (2002), 7821–7826. https://doi.org/10.1073/pnas.122653799 arXiv:https://www.pnas.org/content/99/12/7821.full.pdf

[7] Mohammad Hajian, Emanuel Melachrinoudis, and Peter Kubat. 2016. Modeling wildfire propagation with the stochastic shortest path: A fast simulation approach. *Environmental Modelling Software* 82 (2016), 73 – 88. https://doi.org/10.1016/j.envsoft.2016.03.012

[8] K. Hoover and L. A. Hanson. 2020. Wildfire Statistics. (2020). https://fas.org/sgp/crs/misc/IF10244.pdf

[9] Ioannis Karafyllidis and Adonios Thanailakis. 1997. A model for predicting forest fire spreading using cellular automata. *Ecological Modelling* 99, 1 (1997), 87 – 97. https://doi.org/10.1016/S0304-3800(96)01942-4

[10] Reji Kumar Karunakaran, Shibu Manuel, and Edamana Narayanan Satheesh. 2017. Spreading Information in Complex Networks: An Overview and Some Modified Methods. *Graph Theory-Advanced Algorithms and Applications* (2017).

[11] David Kempe, Jon Kleinberg, and Éva Tardos. 2003. Maximizing the Spread of Influence through a Social Network. In *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (Washington, D.C.) *(KDD '03)*. Association for Computing Machinery, New York, NY, USA, 137–146. https://doi.org/10.1145/956750.956769

[12] Judit Lecina-Diaz, Albert Alvarez, and Javier Retana. 2014. Extreme fire severity patterns in topographic, convective and wind-driven historical wildfires of Mediterranean pine forests. *PloS one* 9, 1 (2014), e85127.

[13] Sang-Woo Lee, Myung-Bo Lee, Young-Geun Lee, Myoung-Soo Won, Jong-Jin Kim, and Sung-kwon Hong. 2009. Relationship between landscape structure and burn severity at the landscape and class levels in Samchuck, South Korea. *Forest Ecology and Management* 258, 7 (2009), 1594–1604.

[14] Trystan Leng, Connor Whie, Joe Hilton, Adam J Kucharski, Lorenzo J Pellis, Helena Stage, Nicholas G Davies, Matt J Keeling, and Stefan Flasche. 2020. The effectiveness of social bubbles as part of a Covid-19 lockdown exit strategy, a modelling study. *medRxiv* (2020).

[15] Jure Leskovec, Anand Rajaraman, and Jeffrey David Ullman. 2014. *Mining of Massive Datasets* (2nd ed.). Cambridge University Press, USA.

[16] Ying Liu, Ming Tang, Tao Zhou, and Younghae Do. 2015. Improving the accuracy of the k-shell method by removing redundant links: From a perspective of spreading dynamics. *Scientific reports* 5 (2015), 13172.

[17] Garrett W Meigs, Christopher J Dunn, Sean A Parks, and Meg A Krawchuk. 2020. Influence of topography and fuels on fire refugia probability under varying fire weather conditions in forests of the Pacific Northwest, USA. *Canadian Journal of Forest Research* 999 (2020), 1–12.

[18] Ralph M Nelson Jr. 2002. An effective wind speed for models of fire spread. *International Journal of Wildland Fire* 11, 2 (2002), 153–161.

[19] José Olabarria, Marc Palahí, Antoni Trasobares, and Timo Pukkala. 2007. A fire probability model for forest stands in Catalonia (north-east Spain). *Annals of Forest Science* 64 (02 2007), 169. https://doi.org/10.1051/forest:2007047

[20] Cristobal Pais, Jaime Carrasco, David L Martell, Andres Weintraub, and David L Woodruff. 2019. Cell2fire: A cell based forest fire growth model. *arXiv preprint arXiv:1905.09317* (2019).

[21] N. Pathak, A. Banerjee, and J. Srivastava. 2010. A Generalized Linear Threshold Model for Multiple Cascades. In *2010 IEEE International Conference on Data Mining*. 965–970. https://doi.org/10.1109/ICDM.2010.153

[22] Marie-Pierre Rogeau and Glen W Armstrong. 2017. Quantifying the effect of elevation and aspect on fire return intervals in the Canadian Rocky Mountains. *Forest Ecology and Management* 384 (2017), 248–261.

[23] R. C. Rothermel. 2017. A Mathematical Model for Predicting Fire Spread in Wildland Fuels.

[24] The USDA Forest Service. 2020. *FSTopo Map Product.* https://data.fs.usda.gov/geodata/rastergateway/states-regions/states.php

[25] Paul A Werth, Brian E Potter, Martin E Alexander, Craig B Clements, Miguel G Cruz, Mark A Finney, Jason M Forthofer, Scott L Goodrick, Chad Hoffman, W Matt Jolly, et al. 2016. Synthesis of knowledge of extreme fire behavior: volume 2 for fire behavior specialists, researchers, and meteorologists. *Gen. Tech. Rep. PNW-GTR-891. Portland, OR: US Department of Agriculture, Forest Service, Pacific Northwest Research Station. 258 p.* 891 (2016).

[26] S. Yassemi, S. Dragićević, and M. Schmidt. 2008. Design and implementation of an integrated GIS-based cellular automata model to characterize forest fire behaviour. *Ecological Modelling* 210, 1 (2008), 71 – 84. https://doi.org/10.1016/j.ecolmodel.2007.07.020

[27] An Zeng and Cheng-Jun Zhang. 2013. Ranking spreaders by decomposing complex networks. *Physics Letters A* 377, 14 (2013), 1031–1035.

APPENDIX

A SUPPLEMENTARY ALGORITHMS

In this section of the appendix, we discuss further in detail some algorithms implemented in the project that did not seem to be relevant towards the problem described.

A.1 The Simulate-Wind Algorithm

In this section, we discuss how wind is simulated during fire simulation. See **Algorithm 4**. We take a graph G as input, along with the user-defined parameter ψ, which should act as the maximum possible wind speed. In Line 1, we pick that wind speed $\gamma \in [0, \psi]$ which will be picked uniformly at random. In Line 2, we pick a forest n in the network at random and we suppose that it has position (n_x, n_y). We then pick in Line 3 a pair of positive integers at random (r_1, r_2). Ideally, we want to pick values within the range of our network lattice. However, we have uplifted this restriction by allowing out-of-bounds results while getting the wind simulation to still working properly. By Line 4, we determine the Area of Effect (AoE). This is simply an elliptical region for where all edges lying in this particular region will experience changes to the wind conditions. At closer inspection the region bounded by Ω seems to be rectangular; however what this area presents is actually elliptical in shape and that only the ranges of the major and minor axes of the ellipse are presented. In Line 5, we store all those edges within the elliptical boundary into the set E_{update}. And then in Line 6 apply those changes to the Graph G, i.e. update the 'wind' attribute of the edges in E_{update} from G with the randomly selected γ value.

Algorithm 4: Simulate-Wind (G, ψ)

Input : Graph G, and a maximum possible speed ψ
Output : (None). The resultant G with wind simulated

1 $\gamma \leftarrow \text{random}[0, \psi]$
2 $n \leftarrow$ pick a random forest, with $pos = (n_x, n_y)$
3 $r_1, r_2 \leftarrow \{a, b \mid a, b \in \text{random}(\mathbb{Z}^+)\}$
4 $\Omega \leftarrow [n_x - r_1, n_x + r_1] \times [n_y - r_2, n_y + r_2]$
5 $E_{\text{update}} \leftarrow \{(n_i, n_j) \mid n_i, n_j \in \Omega\}$
6 $\Delta(G, E_{\text{update}}, \text{'wind'}, \gamma)$

Algorithm 5: Girvan-Newman $(G(N, E), K)$

Input : Graph G, with node set N and edge set E where $|E| \geq 1$, and K is an integer denoting how many edges to remove where $0 < K \leq |E|$
Output : E', the set of removed edges. Also G is now G' where $G' = G(N, E \setminus E')$ and the edges removed are based on edge-betweenness centrality scores

1 Calculate-Edge-Betweenness(G)
2 $E' \leftarrow \{\}$
3 **while** $|E'| < K \wedge E \neq \emptyset$ **do**
4 $\quad e_{\max} \leftarrow$ Mth-Largest-Edge$(G, 1)$
5 $\quad E \leftarrow E \setminus \{e_{\max}\}$; $E' \leftarrow E' \cup \{e_{\max}\}$
6 $\quad$ Calculate-Edge-Betweenness(G)
7 **return** E'

A.2 The Girvan-Newman Algorithm

The Girvan-Newman algorithm is a popular method in detecting communities. We employ this algorithm in the detecting 'forest communities' that we can isolate in our forest network. The algorithm is straight-forward as seen in **Algorithm 5**; we begin by calculating the betweenness of all edges in the forest graph G on line 1. In line 2, we create an empty set E', the set of edges that we intend to remove from G according to the Girvan-Newman heuristic. Lines 3-6 is a loop where we terminate whenever the K edges have already been removed from the graph G or when there are no more edges in the graph to remove. In the loop body, take the edge with largest betweenness score, remove that and add it to the set E'. Recalculate the betweenness scores of all edges in the graph, then repeat the process until termination. In the end at line 7, return the set of edges removed from G, which is E'.

B PROOFS OF THEOREMS

In this section, we discuss the proofs covered in this paper.

B.1 Proof of Theorem 5.1

Proof. ($\Longrightarrow$), the set of nodes of G remains unchanged as we only had removed edges to produce G_F and G_G. Clearly, if $\mathcal{F}$ and $\mathcal{G}$ are the same set of edges, then we get the same graph when removing $\mathcal{F}$ from the set of edges in G and when removing $\mathcal{G}$ from the set of edges in G. Because we are removing the same set of edges, this should also imply that the performance of both intervention algorithms will have to be the same according to how we defined "performance" in **Footnote 6** in page 8. This is due to the deterministic nature of the Linear Threshold Model and the deterministic formula for calculating node thresholds and edge weights.

($\Longleftarrow$), it is clear that if $G_F = G_G$, then the set of edges removed from G to produce G_F have to be same (not necessarily in the same order as we are dealing with sets) as the set of edges from taken out from G to produce G_G. Hence $\mathcal{F} = \mathcal{G}$. □

B.2 Proof of Theorem 5.2

Proof. The **FIGHTER** algorithm simply does not consider the fact that it uses a neighborhood-based edge-removal approach when $\lambda \to \infty$ (as it considers the entire forest space) and simply takes into account the betweenness score factor. Hence this method reduces to GN's approach. Therefore the set of edges **FIGHTER** removes is the same as the set of edges GN will prune (and in the same exact order of selection as well!). Thus $\mathcal{F} = \mathcal{G}$.

The converse is not true, however. If $\mathcal{F} = \mathcal{G}$, this does not say anything about how large our neighborhood-balls must be. It may be possible that λ is small and that $\mathcal{F} = \mathcal{G}$, i.e. the edges as removed by the GN are proximate each other within a distance of λ. □

C ADDITIONAL FIGURES AND TABLES

C.1 Incinerate Forest Fire Simulation

We see in this section an example of the simulation run for **INCINERATE**, as detailed in **Algorithm 1**.

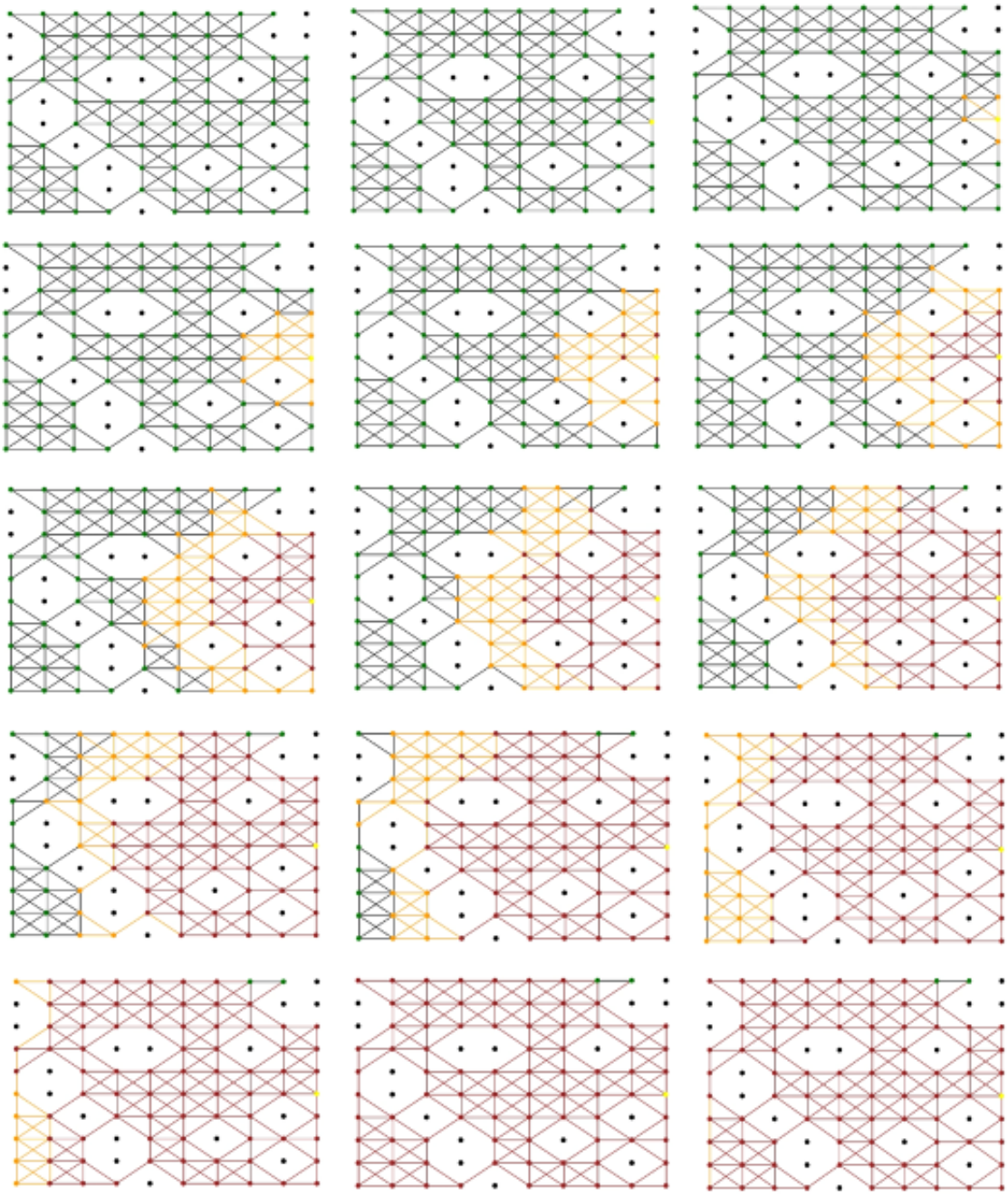


Figure 7: This is a sample run of the forest fire propagation simulation (sequence begins in the upper left corner going across, then down). The figure in the upper left corner panel is the initial state of the system where none of the forests are damaged (all non-burnt forests are green and all those fire zones whose states will never change are black). In the figure beside it shows a randomly selected node in the forest (colored yellow) where it serves as the source of the fire. In the sequences that follow, we see that fire propagates accordingly to our algorithms (where orange nodes represent forests that are burning and brown nodes represent forests that are burnt out). In the final few panels of the simulation, we have seen that the number of burning forests has remained the same and therefore serves as the stopping criterion of the fire propagating algorithm.

C.2 Node-Labelled Forest Network

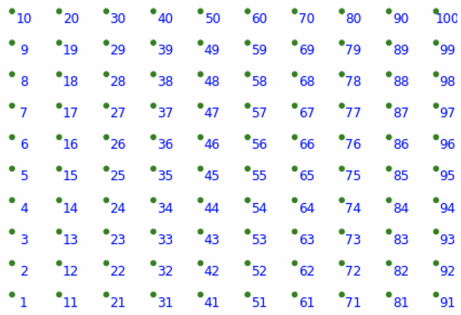


Figure 8: The Forest Network (using 100 nodes), in a square grid lattice. Nodes are labelled sequentially beginning from 1, starting from the bottom up, then left going right until all rows/columns are filled up. Edges are withheld for easier readability of node labels.

C.3 Simulate-Wind Simulation

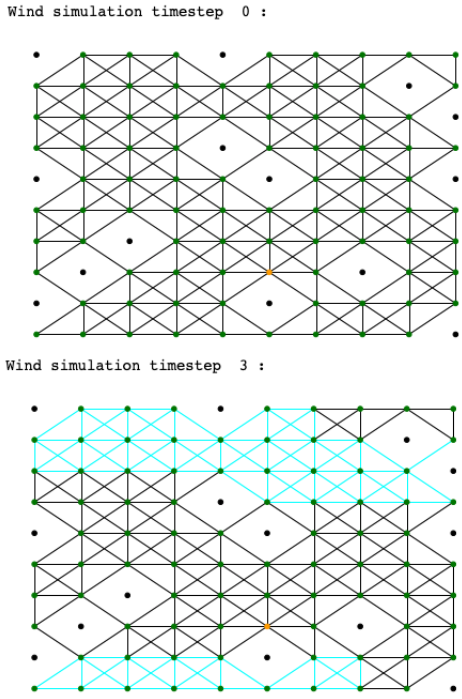


Figure 9: Wind simulation in the Forest Network where wind conditions change at every $t = 1$ time steps (only the initial and final lattice network is shown after 3 timesteps). The edges in cyan color indicate that there were changes with the wind condition at any given time from the initial time of simulation.

We have also explored on the simulation of wind to the forest network, in accordance with **Algorithm 4**. An example of this simulation that runs over 3 time steps can be seen in **Figure 9**. In the simulation for some t timestep, the algorithm chooses a random forest in the network which acts as the center or the "eye" of the area affected by the wind, and nearby edges of the node in the network are considered affected by the change of wind conditions. As one could imagine, this is similar to how wind behaves in a storm or typhoon.

C.4 Aspect Coefficient Lookup Table

We can find the aspect coefficient values from this lookup table, as described by the mathematical model in **Section 4.4**.

Aspect Value A	Direction	Aspect Coefficient α
$A \in [0, 22.5) \cup [337.5, 360]$	North	-0.063
$A \in [22.5, 67.5)$	North-East	0.349
$A \in [67.5, 112.5)$	East	0.686
$A \in [112.5, 157.5)$	South-East	0.557
$A \in [157.5, 202.5)$	South	0.039
$A \in [202.5, 247.5)$	South-West	-0.155
$A \in [247.5, 292.5)$	West	-0.0252
$A \in [292.5, 337.5)$	North-West	-0.171

Table 3: A Lookup Table for the Aspect Coefficient α. Note that the aspect value A can be found under the 'Aspect' attribute in the ForestCoverType dataset [2].